(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 1 225 741 A1

(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
24.07.2002 Bulletin 2002/30

(51) Int Cl.^{7}: **H04L 29/06**

(21) Application number: **00123673.6**

(22) Date of filing: **30.10.2000**

(84) Designated Contracting States:
AT BE CH CY DE DK ES FI FR GB GR IE IT LI LU MC NL PT SE
Designated Extension States:
AL LT LV MK RO SI

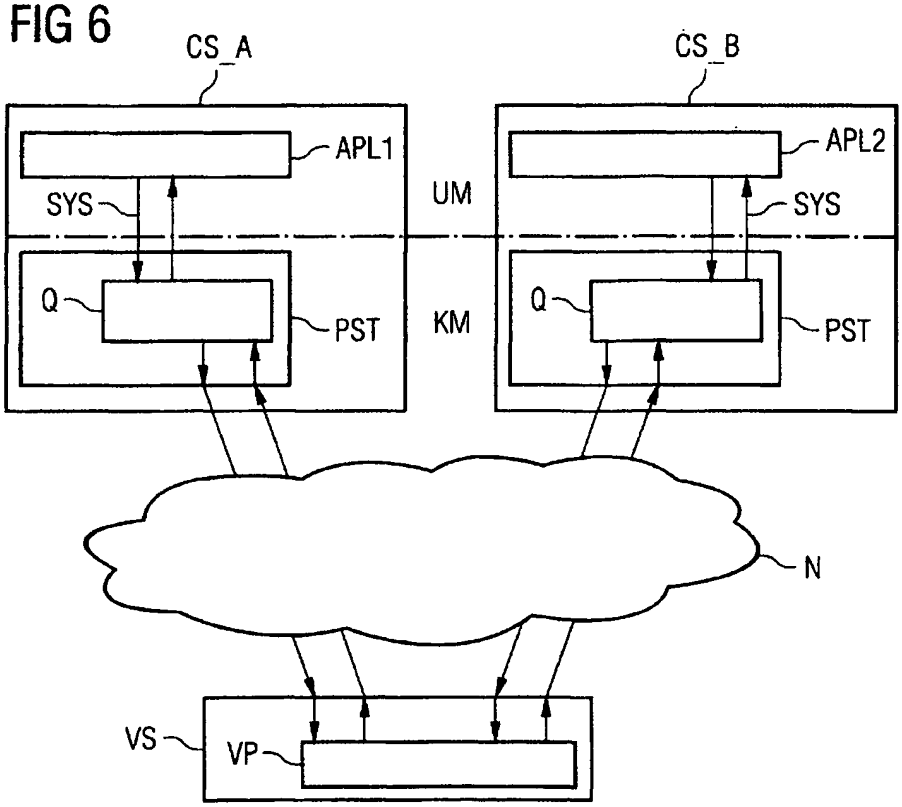
(71) Applicant: **SIEMENS AKTIENGESELLSCHAFT**
80333 München (DE)

(72) Inventors:
- **Birkenmaier, Rainer**
  72516 Scheer (DE)
- **Graebner, Oliver**
  80636 München (DE)
- **Kuefner, Holger, Dr.**
  83026 Rosenheim (DE)

(54) **High speed interconnection for embedded systems within a computer network**

(57) For obtaining an efficient, high-speed and reliable protocol a protocol stack is developed that supports only functionality which is absolutely essential for the application. In particular, it ignores all those functions relating to use in long-distance networks. The inventive step firstly comprises the movement of a distributed synchronization in time of multiple computation systems from the application to the protocol stack for realization of a voting mechanism within the computer network and, secondly, the integration of this functionality in this efficient and reliable protocol.

FIG 6



EP 1 225 741 A1

Description

[0001] The present invention relates to a process for transmitting data from a first system to a second system within a local area computer network using Ethernet communication technology and to a voting mechanism in an error-tolerant system within a computer network making use of such a process for transmitting data, wherein

- a number of different computation systems carry out the same calculation and every result of a calculation from every computation system is transmitted to be compared to a voter system,
- said voter system comparing the different results and reporting a result of the comparison back to said different computation systems involved, wherein

said computation systems do not start any further calculations until they have received the result of the voting process.

[0002] In the field of local Ethernet communication, modern computer networks, in particular in office communication, mainly use network protocols, in particular TCP/IP, which are not optimized for this specific type of communication, but are designed predominantly for long-distance networks.

[0003] The overhead produced in consequence slows down the overall communication process. In addition, special mechanisms have to be modeled at the application level to achieve a synchronization of communications systems. Essentially, this results in three problems (a, b, c) which are to be solved by this invention.

[0004] a) efficient communication, which relates essentially to the response times and the data throughput rate, but also takes account of other areas of the required protocol elements.

[0005] In the past, an improvement in the efficiency of communication has been achieved mainly by using faster hardware. Another approach is to attempt to improve the efficiency by reimplementation of existing protocol stacks or parts of protocol stacks. However, in that case, the fundamental problem still remains that the protocol itself is not optimized for the purpose.

[0006] b) reliable communication, when data are transmitted, they either arrive correctly or a corresponding error is signaled. In either case, the application has thus to be informed of the data transmission status.

[0007] Reliable communication nowadays is either provided by the protocol stack or must be implemented by the application itself.

[0008] c) distributed synchronization of processes for voting mechanisms in error-tolerant systems.

[0009] Distributed synchronization has up until now been achieved only at the application level causing problems with respect to the implementation and the performance.

[0010] An object of the present invention is thus to provide a process for transmitting data within a local area computer network that improves efficiency and reliability of the transmission.

[0011] The aforementioned object is achieved according to the present invention with a process for transmitting data from a first system to a second system within a local area computer network using Ethernet communication technology, wherein existing protocol stacks within local Ethernet communication are redesigned in a way that functions relating to use in long-distance networks, in particular routing capability and/or complex name services such as DNS for TCP/IP and/or multiple checksums, are eliminated by means of

- data packets being identified by a unique identification-number built of an address existing in Ethernet protocol, in particular Medium-Access-Control-addresses, and of a sequence-number, wherein
- received packets are acknowledged to the sending system by acknowledge-IDs, wherein
- a data packet is retransmitted by the sending system on timeout for an acknowledge-ID to be received.

[0012] According to the present invention those functions can be used more efficiently and more easily if they are implemented in the protocol stack itself as described.

[0013] In a first advantageous embodiment of the process for transmitting data according to the present invention it has turned out to be especially advantageous if unique identification-numbers are generated by increasing the identification-number of an actual packet for obtaining a unique identification-number for every following packet.

[0014] In another advantageous embodiment of the process for transmitting data according to the present invention said sending system of a data packet frees memory for this data packet within a data buffer only after having validly received the corresponding acknowledge-ID.

[0015] Under these conditions it has turned out to be especially advantageous if said receiving system of a data packet forgets an identification-number of a data packet only in case of the identification-number being smaller than the one of the last acknowledged acknowledge-ID provided by a sending system.

[0016] Furthermore, it has turned out to be especially advantageous if a timeout for an acknowledge-ID to be received is calculated by means of a

- time a data packet was sent, a
- receive time of the sending system and a
- maximum processing time of the sending system to get a schedule and to start sending a data packet.

[0017] Preferably, the receive time is calculated from

- twice a maximum Ethernet travel time for a data packet
- in addition with a maximum processing time in the receiving system.

[0018] In another advantageous embodiment of the process for transmitting data according to the present invention an error of transmission is detected by a sending system on a number of retransmissions of a data packet exceeding a preset number of attempts.
[0019] Alternatively, an error of transmission is detected by a receiving system on a timeout if an identification-number has to be kept longer than a preset time.
[0020] The advantages achieved with the present invention are especially valuable if every system within the local area computer network is an embedded system.
[0021] In order to handle said acknowledge-IDs efficiently they are transmitted being part of a data packet.
[0022] Under these conditions it has turned out to be especially advantageous if a data packet sent by a sending system also comprises the last acknowledged acknowledge-ID.
[0023] Another object of the invention is to implement a voting mechanism in an error-tolerant system within a computer network making use of the aforementioned process for transmitting data in a simple and efficient manner.
[0024] This object of the present invention is achieved by a voting mechanism in an error-tolerant system within a computer network, wherein

- a number of different computation systems carry out the same calculation and every result of a calculation from every computation system is transmitted to be compared to a voter system,
- said voter system comparing the different results and reporting a result of the comparison back to said different computation systems involved, wherein

said computation systems do not start any further calculations until they have received the result of the voting process,
by the means of a distributed synchronization in time of said different computation systems being implemented as part of a protocol stack for data transmission within the computer network.
[0025] In an advantageous embodiment of the voting mechanism according to the present invention a result of a calculation from a computation system being a requesting system within the network is transmitted to said voter system using a single system call that is not returned until the voting has been completed in the background and a result is present and fed back to said computation system.
[0026] By this feature it is achieved that the handling of a voting process is highly simplified. A single call is sufficient and the process of voting is processed in the background, and is transparent to the application.
[0027] In known voting mechanisms a more critical factor is the fact that this procedure usually forces the application having to wait actively.
[0028] To avoid this problem, in another advantageous embodiment of the voting mechanism according to the present invention a process that issued said system call is placed in a not-ready-queue within said computation system until the system call returns, thus, avoiding said computation system having to wait actively for the result of the voting process.
[0029] It has turned out to be especially advantageous if it is ensured that all the different computation systems involved send the result of the same calculation to the voter system.
[0030] In order to avoid a Single Point off Failure, said voter system comparing the different results and reporting a result of the comparison back to said different computation systems involved is distributed within the computer network.
[0031] Further advantages and details of the present invention are described in the following preferred embodiments together with the drawings. In the drawings, similar elements with the same function are designated with the same reference characters, but similar elements need not necessarily be the same.

FIG 1	shows a block schematic of a protocol stack according to this invention embedded in Windows CE ®,
FIG 2	shows most important parts of a data packed,
FIG 3	shows a time diagram of a data transmission according to the invention,
FIG 4	shows a time diagram of a data transmission according to the invention with a data packet lost,
FIG 5	shows a time diagram of a data transmission according to the invention with an acknowledge-ID lost and
FIG 6	shows a block schematic of a voting mechanism with a system call according to the invention.

[0032] The protocol stack within Ethernet communication in a computer network according to the present invention supports only functionality which is absolutely essential for the application. In particular, it ignores all those functions relating to use in long-distance networks, e.g. routing capability. The implementation is not only matched to local communication, but is also optimized to the communication technology being used, Ethernet. There is no need for any name service, such as DNS for TCP/IP. The Medium-Access-Control addresses MAC from Ethernet are used instead.
[0033] In traditional protocols, such as TCP/IP mechanisms often exist which are used more than once, but whose duplicated utilization does not result in increased

usefulness. For example, this is done in the checksums TCP/IP. These are used to check the correctness of the received data. At the Ethernet level, they are implemented in hardware and are thus virtually free of any latency time. TCP/IP, checksums are calculated again repeatedly at two further levels, which means additional and unnecessary complexity when Ethernet is used as a transport medium.

[0034] The invention thus makes use of an extremely flat and hence highly efficient protocol stack as is shown in FIG 1 which uses the example of Windows CE ® to show how the protocol stack according to the present invention can be integrated in an operating system.

[0035] An application APL is shown that communicates via a protocol stack PST comprising the high speed interconnection protocol I of the present invention and Ethernet communication technology EN for connection with a computer network N.

[0036] FIG 2 shows an example of a data packet P which is used for high speed interconnection according to the invention. Different systems within the computer network N transmit such data packets P for achieving high speed interconnection of those systems in a way that is going to be explained according to FIGs 3 to 5.

[0037] The data packets P as shown in FIG 2 are identified by a unique identification number ID built of the aforementioned MAC Ethernet address that is already in the Ethernet packet and a sequence number seq#. By using the Ethernet address addressing can be realized by making use of the Ethernet socket-interface.

[0038] Usually both communications systems A and B are sending as well as receiving systems. That is why it is advantageous if, as shown in FIG 2, each data packet P, apart from application data, comprises an acknowledge identification number Ack-ID as well as the last acknowledge ID 'last-Ack-ID'. The acknowledge information being part of a data packet makes the interconnection more efficient. Of course, it is also possible to send an acknowledge information in a separate packet to the sending system A.

[0039] For explanation purposes the following description uses communications system A only as sending system and communications system B only as receiving system.

[0040] In FIG 3 to 5 it is thus described how data packets P are transmitted from a first system A being the sending system to a second system B being the receiving system.

[0041] The following time references are used for description:

- t1 time a data packet P is sent
- t2 receive time of a receiving system
- tR receive time of a sending system
- tP1 maximum processing time in a receiving system
- tP2 maximum processing time of a sending system to get a schedule and to start sending
- t3 time the Ack is sent
- t4 time the Ack is received
- tL maximum Ethernet travel time of a data packet transmitted over the computer network N

[0042] FIG 3 shows a transmission of a data packet P working correctly without any packet loss. A communications system A sends 'sd' a data packet P at the timt t1 which is received 'rcv' by a communications system B at the time t2. The received data packet is then processed by the receiving system B within a processing time tP1 before an acknowledge information Ack-ID is sent 'sd ack' back to the communications system A at a time t3 where this acknowledge information is received 'rcv ack' at a time t4. The acknowledge information Ack-ID can be sent being part of a data packet P as described in FIG 2.

[0043] Every communications system A and B can comprise a buffer memory for storing the data of multiple data packets to be transmitted. A part of this buffer memory used for the data packet P having been sent is freed 'del P' only on receiving the correct acknowledge information Ack-ID.

[0044] FIG 4 shows the same way of transmitting data according to the invention but in a case of the data packet P being dropped during transmission to communication system B. As the sending system A does not receive an acknowledge information for the data packet P having been sent 'sd' during a time expected, it has to be retransmitted 'rtm' on timeout.

[0045] A timeout for an acknowledge-ID to be received is calculated by the time t1 the data packet P was sent, the receive time tR of the sending system A and the maximum processing time tP2 of the sending system A to get a schedule and to start sending a data packet according to the following formula:

$$\text{timeout} = t1 + tR + tP2 \quad (1)$$

[0046] Preferably, the receive time tR is calculated from twice a maximum Ethernet travel time tL for a data packet in addition with the maximum processing time tP1 in the receiving system B according to the following formula:

$$tR = 2 * tL + tP1 \quad (2)$$

[0047] Therefore the timeout can be calculated as follows:

$$\text{timeout} = t1 + 2 * tL + tP1 + tP2 \quad (3)$$

[0048] FIG 5 shows the same way of transmitting data as in FIG 3 according to the invention in a case the data packet P having been transmitted correctly but the acknowledge information Ack-ID being dropped during

transmission back to communication system A. As the sending system A does not receive an acknowledge information having been sent 'sd ack' during a time expected, the data packet P again has to be retransmitted 'rtm' on timeout as calculated in formulas (1) to (3).

[0049] When the receiving system B receives 'rcv' the data packet P for the second time it has to resend the acknowledge information Ack-ID but to take into consideration that the data packet P has already been received correctly. By the means of the unique Ids of every data packet the receiving system B is able to recognize that the same data packet P has already been received earlier.

[0050] The time tF a data packet P is received for the second time by the receiving system B according to formula (1) is calculated as follows:

$$tF = t1 + tR + tL + tP2 \qquad (4)$$

[0051] With formula (2) this can be transformed into:

$$tF = t1 + 3 * tL + tP1 + tP2 \qquad (5)$$

[0052] The time t1 is not known by the receiving system B but it can be calculated in worst case as

$$t1 = t2 - tL \qquad (6)$$

so that tF becomes

$$tF = t2 + 2 * tL + tP1 + tP2. \qquad (7)$$

[0053] With

$$t3 = t2 + tP1 \qquad (8)$$

it results the following formula for calculating the time tF of receiving a data packet P for the second time:

$$tF = t3 + 2 * tL + tP2. \qquad (9)$$

[0054] The receiving system B forgets an identification number ID of a data packet P only in case of this ID being smaller than the last acknowledged Ack-ID provided by the sending system A as part of a data packet P according to:

$$ID =< last\text{-}Ack\text{-}ID. \qquad (10)$$

[0055] The sending system A on the one hand recognizes an error of the transmission on too many retransmits. Therefore a number of attempts is preset that must not be exceeded for a correct transmission of data packets P.

[0056] The receiving system B on the other hand recognizes a transmission error on timeout if an ID has to be kept too long.

[0057] An advantage of the data transmission according to the described invention in the field of reliable communication is that datagram communication is protected in the protocol itself as described above in FIGs 3 to 5 in contrast, for example, to TCP/IP. Furthermore, protection mechanisms are used which are matched to the requirements of local communication. Under the conditions described above, on error the complete state of the system can be reconstructed in any case.

[0058] With the high speed and reliable interconnection as described above it is possible to implement a protocol which supports distributed synchronization for voting mechanisms. In a voting mechanism, as it is shown in FIG 6, a number of different computation systems CS_A, CS_B send the result of a calculation which is intended to be compared to a voter system VS, all systems CS_A, CS_B and VS being communications partners within a local computer network N communicating as described above.

[0059] This voter system VS compares the results and reports the result of the comparison back to the different computation systems CS_A, CS_B involved. For this comparison, it is ensured that all the different computation systems CS_A, CS_B involved send the result of the same calculation to the voter system VS, and that they do not start any further calculations until they have received the result of the voting process VP.

[0060] In other words, the different computation systems CS_A, CS_B involved are synchronized in time. In this invention, this voting mechanism is implemented in the protocol stack PST of the different computation systems CS_A, CS_B itself being part of the kernel mode KM or, in other words, of the operating system, e.g. Windows CE ®.

[0061] The applications APL1, APL2 of the different computation systems CS_A, CS_B are located in the user mode UM and communicate with the protocol stack PST including the voting mechanism using a system call SYS of the type 'send back result of voting'.

[0062] Thus, for a synchronization in time no system call SYS is returned until the voting has been completed and a result is present, which is fed back to the different computation systems CS_A, CS_B involved.

[0063] This essentially results in two advantages over an implementation in the application as is state of the art. Firstly, the handling of a voting process VP is highly simplified, a single system call SYS is sufficient and everything else is processed in the background (kernel mode KM), and is transparent to the application. However, a more critical factor is the fact that this procedure avoids the application having to wait actively. Instead of this the requesting process, as is normal for a system

call, is placed in the not-ready queue Q by a process scheduler until the system call SYS returns. The process is thus stopped throughout the entire waiting period, which saves processor time and frees the processor to carryout other processes. Once again, this functionality can be used very easily, by means of a simple function call SYS.

[0064] FIG 6 shows how a typical system function call SYS of the type 'send back result of voting' function takes place. In order to keep the illustration simple and comprehensible, the voter system VS is shown as a single node within the network N although, in fact, it is preferably to be distributed in order to avoid a Single Point off Failure.

[0065] The invention firstly realizes the movement of the distributed synchronization to the protocol stack PST and, secondly, the integration of this functionality in an efficient, high-speed and reliable interconnection protocol I, as it is required in embedded, fault-tolerant systems.

[0066] However, this does not mean that the protocol I can be used only for distributed synchronization. Due to its advantages (efficiency and reliability), it is a valuable alternative to other protocols, even in normal operation.

[0067] The above description of preferred embodiments of the present invention is given for the purpose of illustration, but is not exhaustive. Nor is the present invention limited to the precise form described here, but instead numerous modifications and changes are also possible within the context of the above disclosure.

[0068] Preferred embodiments have been described to illustrate the basic details of the present invention and practical applications to enable those skilled in the art to implement this invention. Numerous additional modifications may be implemented for special applications.

Claims

1. A process for transmitting data from a first system (A) to a second system (B) within a local area computer network (N) using Ethernet (EN) communication technology,
   characterised in that
   for improvement of efficiency existing protocol stacks (PST) within local Ethemet communication are redesigned in a way that functions relating to use in long-distance networks, in particular routing capability and/or complex name services such as DNS for TCP/IP and/or multiple checksums, are eliminated by means of

   - data packets (P) being identified by a unique identification-number (ID) built of an address existing in Ethernet protocol, in particular Medium-Access-Control-addresses (MAC), and of a sequence-number (seq#), wherein
   - received packets (rcv) are acknowledged (ack) to the sending system (A) by acknowledge-IDs (Ack-ID), wherein
   - a data packet (P) is retransmitted (rtm) by the sending system (A) on timeout for an acknowledge-ID (Ack-ID) to be received.

2. Process for transmitting data according to claim 1,
   characterised in that
   unique identification-numbers (ID) are generated by increasing the identification-number (ID) of an actual packet for obtaining a unique identification-number (ID) for every following packet.

3. Process for transmitting data according to claim 1 or 2,
   characterised in that
   the sending system (A) of a data packet (P) frees (del) memory for this data packet (P) within a data buffer only after having validly received (rcv ack) the corresponding Acknowledge-ID (Ack-ID).

4. Process for transmitting data according to claim 2 or 3,
   characterised in that
   the receiving system (B) of a data packet (P) forgets an identification-number (ID) of a data packet (P) only in case of the identification-number (ID) being smaller than the one of the last acknowledged Acknowledge-ID (Ack-ID) provided by a sending system (A).

5. Process for transmitting data according to one of the preceding claims,
   characterised in that
   a timeout for an acknowledge-ID (Ack-ID) to be received is calculated by means of a

   - time (t1) a data packet (P) was sent (sd), a
   - receive time (tR) of the sending system (A) and a
   - maximum processing time (tP2) of the sending system (A) to get a schedule and to start sending (sd) a data packet (P), according to the following formula:

   $$\text{timeout} = t1 + tR + tP2.$$

6. Process for transmitting data according to claim 5,
   characterised in that
   the receive time (tR) for a sending system (A) is calculated from

   - twice a maximum Ethernet travel time (tL) for a data packet (P)
   - in addition with a maximum processing time (tPl) in the receiving system (B).

7. Process for transmitting data according to one of the preceding claims,
characterised in that
an error of transmission is detected by a sending system (A) on a number of retransmissions of a data packet (P) exceeding a preset number of attempts.

8. Process for transmitting data according to one of the preceding claims,
characterised in that
an error of transmission is detected by a receiving system (B) on a timeout if an identification-number (ID) has to be kept longer than a preset time.

9. Process for transmitting data according to one of the preceding claims,
characterised in that
every system (A, B) within the local area computer network (N) is an embedded system.

10. Process for transmitting data according to one of the preceding claims,
characterised in that
acknowledge-IDs (Ack-ID) are transmitted (sd ack) being part of a data packet (P).

11. Process for transmitting data according to one of the preceding claims,
characterised in that
a data packet (P) sent (sd) by a sending system (A) comprises the last acknowledged acknowledge-ID (last-Ack-ID).

12. A voting mechanism in an error-tolerant system within a computer network (N), in particular using a process for transmitting data according to one of the preceding claims, wherein

- a number of different computation systems (CS_A, CS_B) carry out the same calculation and every result of a calculation from every computation system (CS_A, CS_B) is transmitted to be compared to a voter system (VS),
- said voter system (VS) comparing the different results and reporting a result of the comparison back to said different computation systems (CS_A, CS_B) involved, wherein

said computation systems do not start any further calculations until they have received the result of the voting process (VP),
characterised in that
a distributed synchronization in time of said different computation systems (CS_A, CS_B) is implemented as part of a protocol stack (PST) for data transmission within the computer network (N).

13. Voting mechanism according to claim 12,
characterised in that
a result of a calculation from a computation system (CS_A, CS_B) being a requesting system within the network (N) is transmitted to said voter system (VS) using a single system call (SYS) that is not returned until the voting has been completed in the background (KM) and a result is present and fed back to said computation system (CS_A, CS_B).

14. Voting mechanism according to claim 13,
characterised in that
a process that issued said system call (SYS) is placed in a not-ready-queue (Q) within said computation system (CS_A, CS_B) until the system call (SYS) returns, thus, avoiding said computation system (CS_A, CS_B) having to wait actively (UM) for the result of the voting process (VP).

15. Voting mechanism according to claim 12 or 13 or 14,
characterised in that
it is ensured that all the different computation systems (CS_A, CS_B) involved send the result of the same calculation to the voter system (VS).

16. Voting mechanism according to one of the preceding claims 12 to 15,
characterised in that
said voter system (VS) comparing the different results and reporting a result of the comparison back to said different computation systems (CS_A, CS_B) involved is distributed within the computer network (N) in order to avoid a Single Point off Failure.

FIG 1

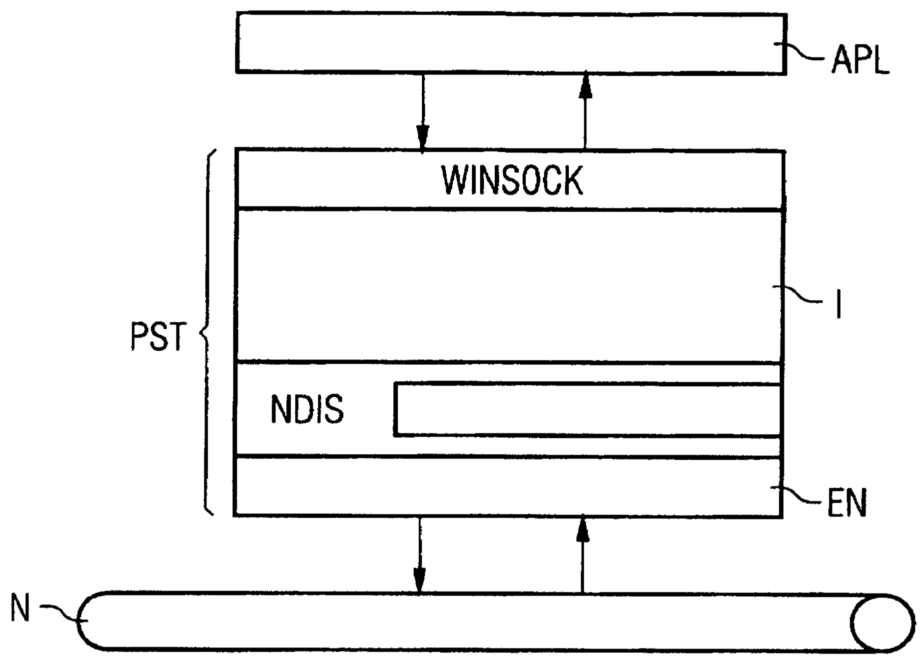


FIG 2

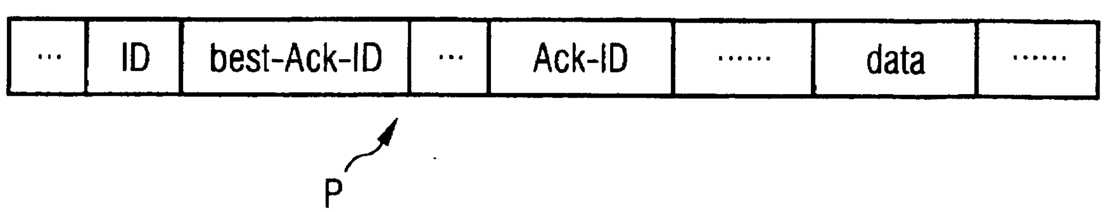


FIG 3

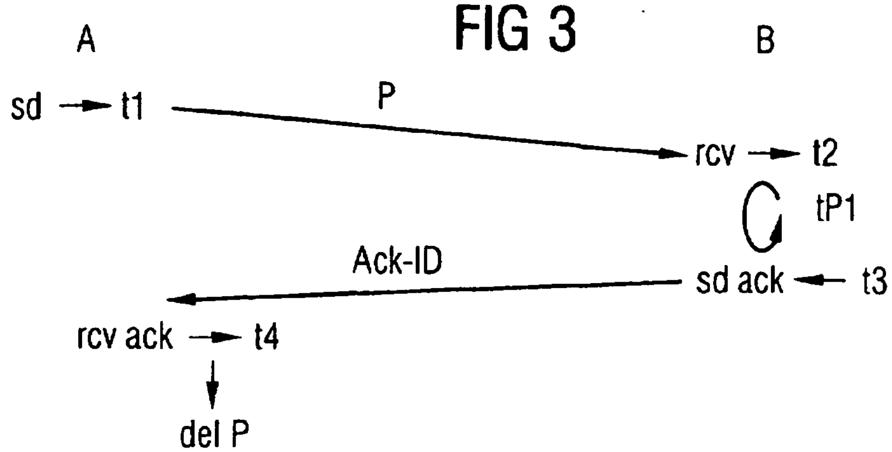


FIG 4

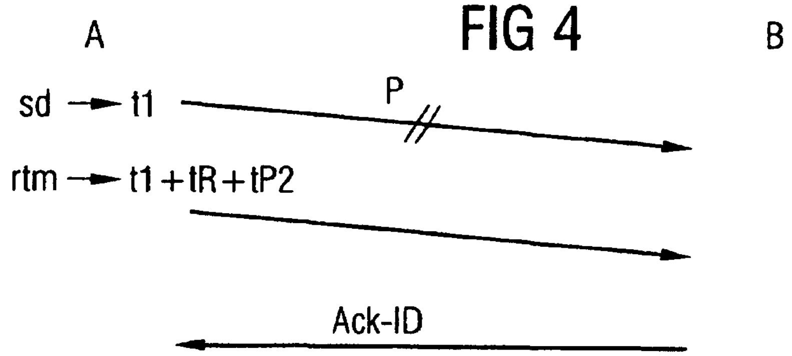


FIG 5

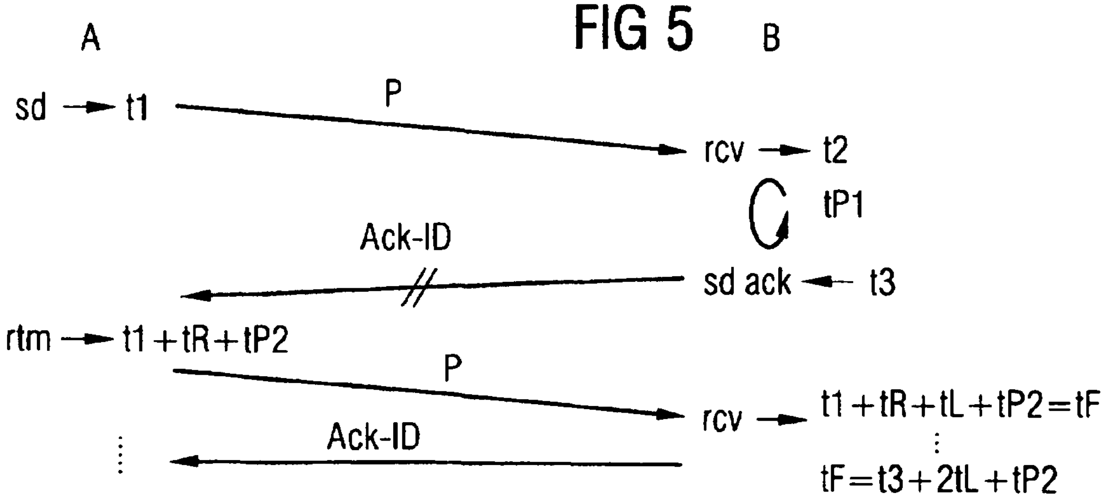
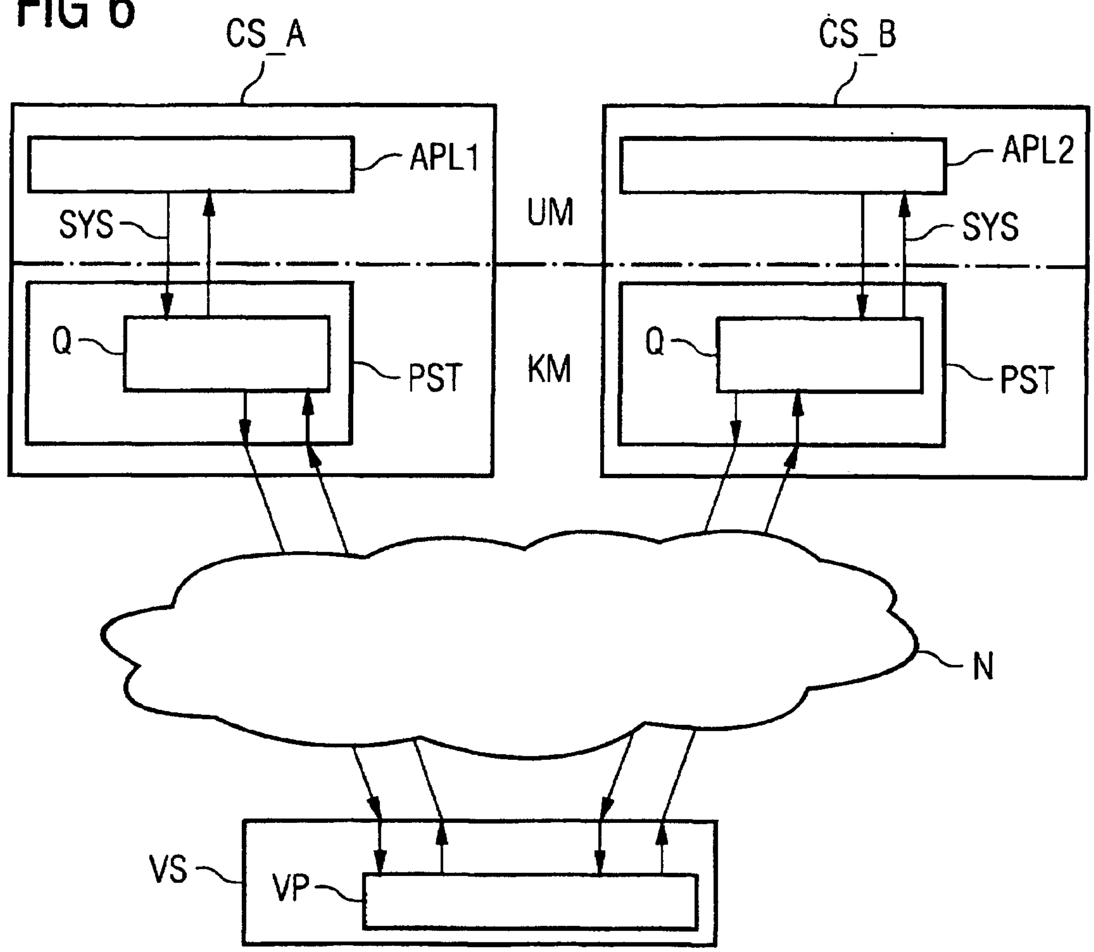


FIG 6



European Patent Office

EUROPEAN SEARCH REPORT

Application Number
EP 00 12 3673

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (Int.Cl.7)
X	WO 00 24165 A (EXCEL SWITCHING CORP) 27 April 2000 (2000-04-27)	1-4,9-11	H04L29/06
A	* page 2, line 17 - page 4, line 4 * * page 10, line 8 - page 15, line 28 *	5,7,8,12	
A	A.S. TANENBAUM: "Computer Networks" 1996 , PRENTICE HALL PTR , UPPER SADDLE RIVER, NEW JERSEY XP002166384 * page 202, paragraph 4 - page 219, paragraph 1 *	1-4,9-11	

TECHNICAL FIELDS SEARCHED (Int.Cl.7)

H04L

The present search report has been drawn up for all claims

Place of search	Date of completion of the search	Examiner
THE HAGUE	2 May 2001	RAMIREZ DE AREL.., F

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04C01)

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 00 12 3673

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report.
The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

02-05-2001

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
WO 0024165 A	27-04-2000	AU 1132400 A	08-05-2000

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82